IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 May 2024, accepted 19 June 2024, date of publication 24 June 2024, date of current version 1 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3418206

RESEARCH ARTICLE

Variable Step Sizes for Iterative Jacobian-Based Inverse Kinematics of Robotic Manipulators

JACINTO COLAN[1], (Member, IEEE), ANA DAVILA[2], (Member, IEEE), AND YASUHISA HASEGAWA[2], (Member, IEEE)

[1]Department of Micro-Nano Mechanical Science and Engineering, Nagoya University, Nagoya 464-8603, Japan
[2]Institutes of Innovation for Future Society, Nagoya University, Nagoya 464-8601, Japan

Corresponding author: Jacinto Colan (colan@robo.mein.nagoya-u.ac.jp)

This work was supported in part by Japan Science and Technology Agency (JST) CREST under Grant JPMJCR20D5, and in part by Japan Society for the Promotion of Science (JSPS) Grants-in-Aid for Scientific Research (KAKENHI) under Grant 22K14221.

ABSTRACT This study evaluates the impact of step size selection on Jacobian-based inverse kinematics (IK) for robotic manipulators. Although traditional constant step size approaches offer simplicity, they often exhibit limitations in convergence speed and performance. To address these challenges, we propose and evaluate novel variable step size strategies. Our work explores three approaches: gradient-based dynamic selection, cyclic alternation, and random sampling techniques. We conducted extensive experiments on various manipulator kinematic chains and IK algorithms to demonstrate the benefits of these approaches. In particular, variable step sizes randomly derived from a normal distribution consistently improve solve rates across all evaluated cases compared to constant step sizes. Incorporating random restarts further enhances performance, effectively mitigating the effect of local minima. Our results suggest that variable step size strategies can improve the performance of Jacobian-based IK methods for robotic manipulators and have potential applications in other nonlinear optimization problems.

INDEX TERMS Robotic manipulators, inverse kinematics, variable step size, random sampling, nonlinear optimization, iterative methods.

I. INTRODUCTION

Robotic manipulators are fundamentally based on kinematic chains that consist of interconnected links and joints that extend from a base to an end effector. The complexity of a manipulator is determined by the number and arrangement of its joints, with a higher degree of freedom (DOF) generally correlated with increased maneuverability and dexterity [1]. For instance, industrial manipulators frequently employ six-DOF serial kinematic chains to achieve precise positioning in a six-dimensional Cartesian space. On the other hand, surgical robots can include more than six DOFs, incorporating a variety of surgical tools to conform to specific operational constraints such as a remote center of motion (RCM) [2]. This variety of kinematic structures in modern manipulators makes the development of a universal kinematics solver for accurate positioning control challenging. Forward kinematics involves mapping a given joint configuration, represented by $\boldsymbol{q} = [q_1, \ldots, q_n]$, to the corresponding end-effector pose $\boldsymbol{\mathcal{X}}_a$. On the contrary, inverse kinematics (IK) aims to determine the joint configuration $\boldsymbol{q}^*$ that results in a desired end-effector pose $\boldsymbol{\mathcal{X}}_d$. This task is essential, as robot operations are typically defined in the operational space (Cartesian space), while the robot is controlled in the joint (configuration) space. Solving the IK problem results in a nonconvex optimization problem that involves scenarios that can yield zero, one, or an infinite number of solutions. The inclusion of joint limits further adds to the complexity, requiring solutions to be constrained within specific ranges.

To address these challenges, several methods have been developed, each with distinct advantages and limitations in terms of computational cost and robustness. Analytic approaches, specific to certain robot geometries, use closed-form solutions for fast computation [3]. However, these methods are limited in their applicability and often lack the flexibility required for additional constraints. Metaheuristic

The associate editor coordinating the review of this manuscript and approving it for publication was Min Wang.

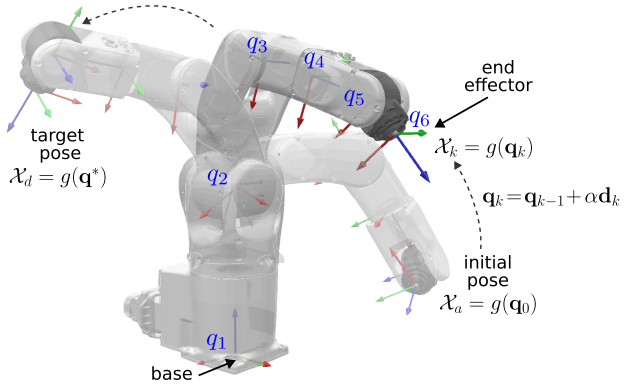

FIGURE 1. Illustration of iterative Jacobian-based inverse kinematics for a robotic manipulator.

approaches, utilizing population-based search algorithms, are appropriate for complex high-dimensional problems [4], but can be computationally demanding and less efficient in scenarios requiring high accuracy. Optimization-based methods approach the problem as a constrained optimization task, effectively managing joint limits and other constraints, but often involve a high computational cost [5], [6], [7]. Numerical methods, which apply iterative optimization techniques such as Jacobian pseudoinverse, numerically derive solutions [8]. Although these methods are computationally intensive, they offer a broader range of applicability compared to analytic methods and are generally faster and more efficient than optimization-based approaches. Hybrid approaches that combine numerical and optimization techniques have been introduced, offering improvements in computation times and solving rates [9], [10], [11]. However, achieving optimal synchronization among different solvers remains an important challenge, accompanied by significant computational demands.

Among the numerical approaches to solving inverse kinematics, iterative descent techniques guided by the Jacobian matrix of the manipulator are commonly employed [12]. These methods take advantage of iterative refinements of an initial joint configuration, $\boldsymbol{q}_0$, to guide the manipulator toward the desired configuration. Several methods in this category have been proposed, including the Jacobian transpose [13], Jacobian pseudoinverse [14], damped least squares [15], and other variations [16], [17]. The primary objective of these methods is to iteratively minimize the error between the current and the desired end-effector poses, aiming to converge to a configuration $\boldsymbol{q}^*$ that achieves the target pose $\boldsymbol{\mathcal{X}}_d$, as illustrated in Fig. 1. Iterative optimization generally involves two main steps at each iteration,

1) Finding a descent direction $\boldsymbol{d}$: This involves identifying a direction that reduces the error between the current configuration and the desired pose.
2) Choosing a step size α: This involves determining the magnitude of the step along the identified descent direction. Finding the optimal step size balances efficient progress toward the desired pose, avoiding overshooting the target or causing system instability.

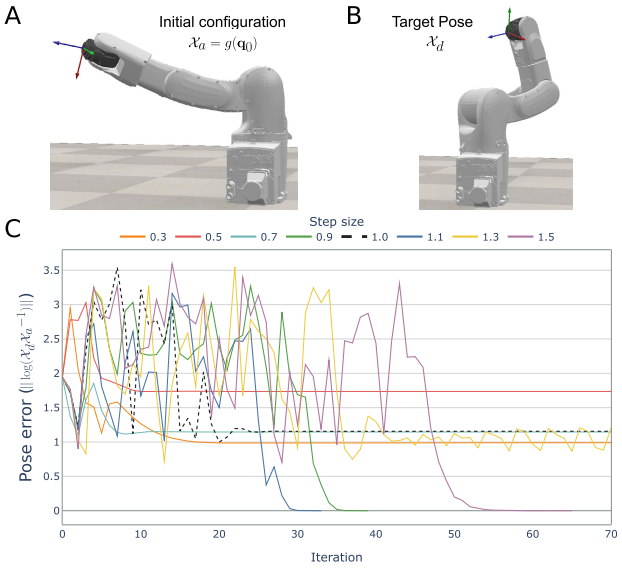

FIGURE 2. Sensitivity of pose error convergence to different constant step sizes in a 6-DOF IK optimization. A. Initial manipulator configuration. B. Target end-effector pose. C. End-effector pose error progression across optimization iterations.

Although choosing the descent direction in Jacobian-based IK follows well-established gradient-based schemes, selecting the step size presents a nuanced challenge. Achieving a balance between making progress towards the solution and maintaining stability is essential for optimal performance. A large step size might induce oscillations and lead to failure, while a small step size necessitates more iterations and increases the risk of encountering local minima. Traditional analyses often assume a constant step size schedule [18], but previous studies suggest that this choice plays a key role in performance rather than the direction of the gradient itself [19]. In the context of Jacobian-based inverse kinematics, this often requires manual adjustment for specific kinematic chains. This results in step sizes effective within the tested configurations but cannot be universally guaranteed across all robotic workspaces.

To illustrate the importance of step size selection, consider a 6-DOF serial robotic manipulator with an initial joint configuration as shown in Fig. 2A, aiming to reach the target pose depicted in Fig. 2B. Using an iterative pseudoinverse IK approach [20], the joint updates at each iteration $\Delta \boldsymbol{q}$ are calculated as $\alpha \boldsymbol{J}^{\dagger} \boldsymbol{e}$, where $\boldsymbol{J}$ denotes the Jacobian matrix of the manipulator and $\boldsymbol{e}$ represents the difference between $\boldsymbol{\mathcal{X}}_{a_k}$, the actual end-effector pose at iteration k, and the target pose $\boldsymbol{\mathcal{X}}_d$. Fig. 2C illustrates the error progression. Employing a conventional constant step size $\alpha^{\text{FIXED}} = 1.0$, represented by the black dashed line, does not reach the convergence threshold. In contrast, employing a slightly different value of α, such as $\alpha = 0.9$ or $\alpha = 1.1$, achieves convergence in fewer than 50 iterations, highlighting the sensitivity of IK optimization to step size selection.

However, it is well established that a constant step size, even when optimized, can result in slow convergence [21]. Variable step size strategies have been proposed to provide dynamic update of the step size at each iteration. A prevalent technique for step size selection in gradient descent optimization involves a line search, in which the step size α is determined as a minimization problem of the objective function F, given by:

$$\alpha = \arg\min_{\alpha>0}\{F(\boldsymbol{q}+\alpha\boldsymbol{d})\} \tag{1}$$

Line search methods require additional gradient evaluations to dynamically adjust the step size, adapting it based on the function's behavior near the current iterate. These methods are designed to ensure a sufficient reduction in the function value at each iteration, forming the basis for techniques such as the Cauchy and Barzilai-Borwein methods [22], [23] and their various adaptations [24], [25], [26], [27], [28], [29]. Backtracking line search is a variation of this method [30], [31], which starts with a large step size and then reduces it until it satisfies a specific criterion, such as the Armijo-Goldstein condition [32]. Traditional backtracking methods only decrease the step size, while adaptive variants can adjust it in both directions in each iteration [33], [34], [35]. However, these methods either limit the increase in step size to a constant factor per iteration or require extra gradient evaluations for step size determination. In large-scale gradient-based optimization, particularly in machine learning applications, there has been a shift towards the development of self-adaptive strategies to adjust step sizes, also known as learning rates, based on gradient changes in each iteration [36], [37]. Recent research in this domain has also investigated the potential of cyclic alternation [21], [38], [39] and random sampling [40], [41] to select dynamic step sizes in nonlinear optimization.

Despite these advances, the efficacy of each method is strongly influenced by the specific nonlinearities inherent to the problem. Inverse kinematics poses a distinct challenge, characterized by its highly nonlinear relationship between joint and operational space variables. Exploration into the use of variable step sizes in this domain has been limited. For example, studies by Wang et al. [42], [43] have investigated the use of deep neural networks and machine learning models such as Decision Trees, Random Forest, and K-Neighbors, to predict optimal step sizes for the Gaussian damped least squares method in redundant manipulator IK. However, reliance on kinematic chain-specific training reduces the applicability of their approach. In addition, their focus on position targets, excluding orientation considerations, simplifies the optimization problem.

To address the limitations of constant step size methods in Jacobian-based inverse kinematics (IK) for robotic manipulators, this study introduces a novel approach that combines traditional iterative Jacobian methods with variable step sizes. While variable step size strategies are well-documented in general optimization problems, their application and comprehensive evaluation in robotic inverse kinematics have not been thoroughly explored. This research conducts a detailed comparative analysis of various variable step size strategies, incorporating both conventional methods and innovative approaches from fields such as machine learning, specifically applied to inverse kinematics, a context in which these strategies have not been previously evaluated. The effectiveness of these strategies is assessed across different robot kinematic chains and inverse Jacobian IK methods, providing a unique and extensive examination not found in previous works. The key contributions of this research are:

- A comprehensive review and evaluation of self-adaptive strategies to adjust the size of steps in the inverse kinematics of robotic manipulators.
- The development and implementation of diverse step size strategies, such as gradient-based dynamic selection, cyclic alternation, and random sampling, to determine their impact on the efficiency of Jacobian-based IK.
- Extensive experimental validation across various manipulator kinematic chains and IK methods, demonstrating the superiority of variable step size selection over traditional constant step size methods.

The paper is organized as follows. In Section II, we provide a comprehensive overview of Jacobian-based inverse kinematics and the proposed variable step size methods. Section III details the experimental results and performance evaluations, including a comparative analysis of the solving effectiveness of various established IK methodologies. Section IV concludes the paper, summarizing key findings and suggesting avenues for future research in this domain.

II. METHODOLOGY

A. JACOBIAN-BASED INVERSE KINEMATICS

This section briefly reviews the fundamentals of Jacobian-based inverse kinematics. Consider a robotic manipulator kinematic chain composed of n joints. The inverse kinematics problem (IKP) is defined as the inverse of the forward kinematics function $\boldsymbol{g} : \mathbb{R}^n \rightarrow \mathrm{SE}(3)$, which maps a joint configuration $\boldsymbol{q} \in \mathbb{R}^n$ to the end-effector pose $\boldsymbol{\mathcal{X}} \in \mathrm{SE}(3)$, where SE(3) denotes the special Euclidean group. The IKP can be formulated as a nonlinear least squares problem (NLSP), where the optimal joint configuration q^* is obtained by solving the following minimization problem:

$$\boldsymbol{q}^* = \arg\min_{\boldsymbol{q}\in\mathbb{R}^n}\{F(\boldsymbol{q})\} \tag{2}$$

The objective function $F(\mathbf{q})$ is defined as

$$F(\boldsymbol{q}) = \frac{1}{2}||\boldsymbol{r}(\boldsymbol{q})||_2^2 = \frac{1}{2}\boldsymbol{r}(\boldsymbol{q})^\top\boldsymbol{r}(\boldsymbol{q}) \tag{3}$$

where $\boldsymbol{r}(\boldsymbol{q}) = \boldsymbol{\mathcal{X}}_d - \boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^m$ represents the residual vector function, quantifying the discrepancy between the end-effector target pose $\boldsymbol{\mathcal{X}}_d \in \mathrm{SE}(3)$ and the actual pose derived from forward kinematics $\boldsymbol{g}(\boldsymbol{q})$. To improve readability, subsequent references will omit the explicit dependency on $\mathbf{q}$.

Iterative descent methods are frequently used in the solution of nonlinear least squares problems (NLSPs) due to their proven effectiveness, efficiency, and adaptability to various types of problems [44]. These methods begin with the selection of an initial solution guess $\boldsymbol{q}_0$. The objective is to find the minimum of the function F by iteratively moving in a direction of descent, which is based on the gradient and is denoted as $\boldsymbol{d}_k$. The update of the solution in each iteration involves a step size α, as expressed by the following equation:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \alpha \mathbf{d}_k \tag{4}$$

where k represents the iteration index. Two prevalent approaches for determining a descent direction are Gradient Descent (GD) and Newton's Method (NM). GD is a widely used minimization method that operates on the first-order approximation of the objective function. It iteratively moves towards the steepest descent direction, indicated by the negative of the gradient of F, resulting in $\boldsymbol{d} = -\nabla F \in \mathbb{R}^n$. From (3), the gradient can be defined as

$$\nabla F = -\boldsymbol{J}^\top \boldsymbol{r} \tag{5}$$

where $\boldsymbol{J} = \frac{\delta \boldsymbol{g}(\boldsymbol{q})}{\delta \boldsymbol{q}} \in \mathbb{R}^{m \times n}$ represents the Jacobian matrix of the manipulator. The application of this update rule in the context of inverse kinematics results in the well-known Jacobian transpose inverse kinematics method [13], as expressed in the following equation:

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - \alpha \nabla F_k = \boldsymbol{q}_k + \alpha \boldsymbol{J}_k^\top \boldsymbol{r}_k \tag{6}$$

NM, in contrast, leverages the second-order information of the objective function, encapsulated in the Hessian matrix. This approach typically results in a faster convergence rate compared to GD but requires more computational resources due to the necessity of inverting the Hessian matrix in each iteration. The Hessian matrix for the manipulator, $\nabla^2 F$, is defined as [45]

$$\nabla^2 F = \boldsymbol{J}^\top \boldsymbol{J} - \sum_{i=1}^{n} \frac{\delta \boldsymbol{J}}{\delta q_i}^\top \boldsymbol{r} \tag{7}$$

.

The update rule for NM in the context of inverse kinematics is given by

$$\begin{aligned} \boldsymbol{q}_{k+1} &= \boldsymbol{q}_k - {\nabla^2 F_k}^{-1} \nabla F_k \\ &= \boldsymbol{q}_k + \left(\boldsymbol{J}_k^\top \boldsymbol{J}_k - \sum_{i=1}^{n} \frac{\delta \boldsymbol{J}_k}{\delta q_i}^\top \boldsymbol{r}_k \right)^{-1} \boldsymbol{J}_k^\top \boldsymbol{r}_k \end{aligned} \tag{8}$$

Although Newton's method offers faster convergence, the computational expense of calculating the Hessian matrix can be considerable. Moreover, its convergence is not always guaranteed since the Hessian may not be positive definite in every scenario. These challenges have led to the development of various adaptations, among which the Gauss-Newton (GN) method is notably prevalent [46]. The GN method represents an efficient strategy, based on a linear model of $\boldsymbol{r}_k$ around $\boldsymbol{q}_k$, that simplifies the calculation of the Hessian by assuming that the second-order derivatives are negligible. Consequently, the Hessian is approximated by $\nabla^2 F \approx J^\top J$, markedly reducing the computational load compared to the full Hessian calculation. The descent direction $\boldsymbol{d}_k$ in the GN method is determined by solving the following linear equation:

$$\boldsymbol{J}_k^\top \boldsymbol{J}_k \boldsymbol{d}_k = \boldsymbol{J}_k^\top \boldsymbol{r}_k. \tag{9}$$

This corresponds to addressing the linearized least squares problem

$$\underset{\Delta \boldsymbol{q} \in \mathbb{R}^n}{\arg\min} \frac{1}{2} ||\boldsymbol{J}_k \boldsymbol{d}_k + \boldsymbol{r}_k||_2^2 \tag{10}$$

Subsequently, the joint angle vector $\mathbf{q}_k$ is updated in each iteration as follows:

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - \left(\boldsymbol{J}_k^\top \boldsymbol{J}_k \right)^{-1} \boldsymbol{J}_k^\top \boldsymbol{r}_k \tag{11}$$

The matrix $\boldsymbol{J}^\dagger = (\boldsymbol{J}^\top \boldsymbol{J})^{-1} \boldsymbol{J}^\top$ is recognized as the Moore-Penrose pseudoinverse of $\mathbf{J}$. However, the Gauss-Newton (GN) method does not always guarantee convergence, especially when the initial point is far from the optimum. An improved version of the Gauss-Newton method applies a fraction α of the direction vector to ensure that $f(q)$ decreases in every iteration, thereby improving convergence. This can be expressed as an adaptation of the pseudoinverse method for inverse kinematics,

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \alpha \boldsymbol{J}^\dagger \boldsymbol{r}_k \tag{12}$$

where α represents the size of the step along the descent direction.

The pseudoinverse approach is widely utilized in inverse kinematics due to its ability to handle redundancy and non-square Jacobian matrices. However, it can be problematic in scenarios involving ill-conditioned systems or singularities, where it may produce large joint displacements or fail to converge. To address these issues, variations of the Gauss-Newton method that incorporate damping factors have been proposed, which can improve the stability and robustness of the inverse kinematics solution. Damping least squares methods can be formulated by introducing the damping factor $\lambda \boldsymbol{I}$ into the Hessian approximation, as shown in the following equation:

$$\boldsymbol{q}_{k+1} = \boldsymbol{q}_k - \alpha \left(J_k^\top J_k + \lambda \boldsymbol{I} \right)^{-1} J_k^\top \boldsymbol{r}_k \tag{13}$$

where λ corresponds to the damping factor. The selection of an appropriate damping factor is crucial to ensure convergence, and several methods for its determination have been proposed in the literatures [45] and [47]. The Levenberg-Marquardt (LM) approach is a notable example, which employs a damping factor to constrain the step within an elliptical trust region, ensuring the positive definiteness of the Hessian approximation.

B. VARIABLE STEP SIZE METHODS

Variable step size strategies are based on the principle that different regions within the solution space require varying levels of precision and granularity in step size. As one approaches the solution, it is feasible to employ larger steps for a rapid traversal of the space. Conversely, as one draws closer to the solution, smaller steps become advantageous to refining the approach and preventing overshoot. This method yields two main benefits:

- Convergence Speed: Employing variable step sizes can markedly speed up convergence by enabling the algorithm to switch between larger steps in less sensitive regions of the solution space, thus covering larger distances swiftly, and smaller steps in regions where accuracy is crucial to ensure the optimal solution is achieved.
- Robustness Against Local Minima: The flexibility in adjusting step sizes enhances the stability of inverse kinematics solutions by helping the algorithm avoid local minima. Larger step sizes can provide the needed momentum to overcome the influence of a local minimum, while smaller step sizes can prevent getting stuck initially.

This section reviews established variable step size strategies, predominantly those based on line search techniques, and introduces emerging approaches that leverage gradient-based dynamic selection, cyclic alternation, and random sampling.

1) LINE SEARCH STEP SIZE METHODS

Line search methods are widely used to determine the step size α in gradient descent (GD) and Gauss-Newton (GN) methods. These methods aim to find a step size value that reduces the objective function along the current search direction. However, they do not necessarily guarantee convergence to a global minimum; they only ensure a reduction in the function value, which may lead to convergence to a local rather than a global minimum, especially if the objective function possesses multiple local minima.

For convex quadratic functions of the form $f(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{q}^\top \boldsymbol{A}\boldsymbol{q} - \boldsymbol{b}^\top \boldsymbol{q}$, several step size criteria have been proposed based on the line search approach. The steepest descent (SD) method, originally proposed by Cauchy [22], defines the step size by minimizing the univariate function presented in (1) and has the form

$$\alpha_k^{SD} = \arg\min_{\alpha>0} f(\boldsymbol{q}_k - \alpha \boldsymbol{d}_k) = \frac{\boldsymbol{d}_k{}^\top \boldsymbol{d}_k}{\boldsymbol{d}_k{}^\top \boldsymbol{A}\boldsymbol{d}_k} \tag{14}$$

where $\boldsymbol{d}_k$ corresponds to the gradient at the k-th iteration of $f(\boldsymbol{q})$, $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, and α_k^{SD} represents the step size. Despite its theoretical basis, the SD method often shows slow convergence and is adversely affected by ill-conditioning [48]. Barzilai and Borwein [23] introduced two efficient methods to determine the size of the step, known as BB methods, defined as follows:

$$\begin{aligned}\alpha_k^{BB1} &= \arg\min_{\alpha>0} ||(\alpha^{-1}\boldsymbol{I})\boldsymbol{s}_{k-1} - \boldsymbol{y}_{k-1}|| \\ &= \frac{\boldsymbol{s}_{k-1}^\top \boldsymbol{s}_{k-1}}{\boldsymbol{s}_{k-1}^\top \boldsymbol{y}_{k-1}} = \frac{\boldsymbol{d}_{k-1}{}^\top \boldsymbol{d}_{k-1}}{\boldsymbol{d}_{k-1}{}^\top \boldsymbol{A}\boldsymbol{d}_{k-1}}\end{aligned} \tag{15}$$

$$\begin{aligned}\alpha_k^{BB2} &= \arg\min_{\alpha>0} ||\boldsymbol{s}_{k-1} - (\alpha\boldsymbol{I})\boldsymbol{y}_{k-1}|| \\ &= \frac{\boldsymbol{s}_{k-1}^\top \boldsymbol{y}_{k-1}}{\boldsymbol{y}_{k-1}^\top \boldsymbol{y}_{k-1}} = \frac{\boldsymbol{d}_{k-1}{}^\top \boldsymbol{A}\boldsymbol{d}_{k-1}}{\boldsymbol{d}_{k-1}{}^\top \boldsymbol{A}^2\boldsymbol{d}_{k-1}}\end{aligned} \tag{16}$$

where $\boldsymbol{s}_{k-1} = \boldsymbol{q}_k - \boldsymbol{q}_{k-1}$, and $\boldsymbol{y}_{k-1} = \boldsymbol{d}_k - \boldsymbol{d}_{k-1}$.

Line search is also used to determine the step size in the improved Gauss-Newton method, following either direct search or backtracking approaches. In the direct search method, the optimal step size is obtained by minimizing the function described in (1). However, direct search can be computationally intensive, as it requires solving an additional optimization problem, which is not always straightforward. Backtracking line search methods follow a systematic and deterministic approach to find a suitably large step size. It starts with an initial guess for the step size α_0 and iteratively reduces it until a condition is satisfied, such as the Armijo-Goldstein condition given by

$$f(\boldsymbol{q}_k + \alpha_k \boldsymbol{d}_k) \leq f_k - \mu\alpha \nabla f_k{}^\top \boldsymbol{d}_k \tag{17}$$

for a given constant $\mu \in [0, 1]$.

2) GRADIENT-BASED DYNAMIC SELECTION

Automatic selection of the step size, also known as learning rates, has been a recent development in large-scale optimization, such as training of deep neural network architectures. Gradient scaling (GS) is a technique that has shown promising results in automatically selecting learning rates during the training of deep fully connected and convolutional networks, eliminating the need for additional hyperparameters [36]. In the context of IKP, it scales the gradient component for each joint i by the norm of the gradient vector,

$$\alpha_{k_i}^{GS} = \frac{|\nabla f_{k_i}|}{||\nabla f_k||_2} \quad \forall i \in [1, \ldots, n] \tag{18}$$

A second approach is the Relative Gradient Norm (RGN). This method, initially proposed for transfer learning applications in fine-tuning pre-trained architectures by adjusting the learning rate of model layers [37], [49], is adapted here for the inverse kinematics (IK) problem as discussed in [50]. It involves a two-step scaling process. Initially, the relative gradient φ_k in iteration k is calculated as the relationship between the gradient component of each joint and the value of the joint:

$$\varphi_{k_i} = \left| \frac{\nabla f_{k_i}}{q_i} \right|. \tag{19}$$

Subsequently, each component of φ_k is scaled by its norm:

$$\alpha_{k_i}^{RGN} = \frac{\varphi_{k_i}}{||\varphi_k||_2} \quad \forall i \in [1, \ldots, n] \tag{20}$$

The underlying hypothesis is that smaller relative gradient values indicate a minimal impact of the corresponding joint in reducing pose error. Consequently, the step sizes for these joints can be reduced to prevent interference with other joints that may exhibit larger relative gradients and thus require larger steps for convergence. These adaptive approaches aim to prevent overshooting the optimal values and facilitate faster convergence.

3) CYCLIC ALTERNATION

Cyclic alternation of step sizes introduces a sequence of varying step magnitudes, strategically alternating between shorter and larger steps, to improve the convergence traits of optimization algorithms. This approach diverges from constant step size methodologies by injecting variability into the step size determination process.

Several methodologies have been developed to establish these cyclic sequences. Goujaud et al. [38] develop a convergence rate analysis for quadratic objectives that identifies optimal parameters, showing that cyclical learning rates can effectively exceed the limitations of traditional methods. Altschuler and Parrilo [21] proposed a novel step size schedule for gradient descent that can achieve faster convergence in smooth convex optimization, by hedging between short and long steps in a fractal-like pattern. In contrast with traditional GD analyses for selecting step size, such as line search, this approach focuses on the aggregate advancement made by all iterations rather than individual step progress and shows that properly combining suboptimal step sizes yields faster convergence due to the misalignment of worst-case functions, where bad cases for a long step are good cases for a short step, and vice versa. Grimmer [39], on the other hand, explored policies that incorporate non-constant step sizes with frequent, longer steps that might temporarily increase the objective value but ultimately facilitate faster convergence when evaluated across multiple iterations.

The formulation of cyclic step sizes can be represented as a continuous function $S(k)$ or as a discrete set $S = \{s_0, s_1, \ldots, s_{N-1}\}$, where N denotes the cycle length. In discrete cyclic alternation, the step size for each iteration k is defined as:

$$\alpha_k^{ALT} = s_{(k \bmod N)} \tag{21}$$

In this model, the set S can be selected according to specific criteria or adaptively tuned according to the performance of the function at each iteration.

4) RANDOM SAMPLING STEP SIZE

Random search has been proposed for numerical optimization of unconstrained objectives and has demonstrated its effectiveness in a variety of ill-structured global optimization problems involving both continuous and discrete variables [51]. Random search algorithms typically focus on quickly finding a satisfactory solution, rather than ensuring optimality with probabilistic convergence. Schumer and Steiglitz [40] introduced an adaptive step size random search, where the step size is dynamically adjusted based on the success or failure of the search process. Schrack and Choit [52] suggested calculating the step size as the optimum along a randomly chosen direction vector. More recently, randomized step sizes have been applied to neural network optimization, where each unit or feature of the network is assigned a learning rate sampled from a wide-ranging distribution [41]. Unlike previous work, this study proposes a combination of conventional inverse kinematics (IK) methods to determine a descent direction $\mathbf{d}_k$ and randomly samples step sizes from a probability distribution $\mathcal{P}$ with parameters θ as given by

$$\alpha^{\text{RAND}} \sim \mathcal{P}(\theta) \tag{22}$$

This approach does not adhere to a monotonic decrease in the objective function. The rationale behind this approach is that minor deviations from the conventional constant step size ($\alpha = 1.0$) can effectively counter the principal challenge in numerical optimization, namely getting trapped in local minima, a frequent issue in IKP when the target pose is substantially distant from the current configuration. Empirical evidence is presented to show that random step sizes can significantly improve the solve rate for nonlinear IKP in robotic manipulators. Two types of sampling distribution are evaluated: uniform and normal distributions.

III. EXPERIMENTS AND RESULTS

A. JACOBIAN-BASED FORMULATION FOR INVERSE KINEMATICS PROBLEM

Actual and desired end-effector poses are defined as transformations $\mathbf{X}_a = g(\mathbf{q}_0) \in \text{SE}(3)$ and $\mathbf{X}_d \in \text{SE}(3)$, respectively. The residual in the inverse kinematics problem, as defined in (3), is expressed as:

$$\mathbf{r}(\mathbf{q}) = \log\left(\mathbf{X}_d \mathbf{X}_a{}^{-1}\right) \tag{23}$$

where the logarithm function $\log : \text{SE}(3) \to \mathfrak{se}(3)$ maps the pose from the Lie group SE(3) to twists in the Lie algebra $\mathfrak{se}(3)$ [53]. The goal of the IKP is to find an appropriate joint configuration $\mathbf{q}^*$ that minimizes:

$$\underset{\mathbf{q}\in\mathbb{R}^n}{\arg\min} \frac{1}{2}\left\|\log\left(\mathbf{X}_d \mathbf{X}_a{}^{-1}\right)\right\|_2^2 \tag{24}$$

The optimal $\mathbf{q}^*$ is obtained through a gradient-based iterative optimization process defined in (4), comprising a descent direction $\mathbf{d}_k$ computed from the IK method and a step size α_k for each iteration. The process repeats until the convergence criterion $\|\mathbf{r}(\mathbf{q})\| \leq \epsilon_c$ is satisfied, where ϵ_c denotes the convergence threshold.

Five Jacobian-based inverse kinematics algorithms were implemented for experimental evaluation.

- Jacobian pseudoinverse (PINV) [14]:

$$\boldsymbol{d}_k = \boldsymbol{J}_k^{\dagger} \boldsymbol{r}_k \tag{25}$$

- Damped least squares (DLS) [15]:

$$\boldsymbol{d}_k = (\boldsymbol{J}_k^\top \boldsymbol{J}_k + \lambda_{DLS}\boldsymbol{I})^{-1}\boldsymbol{J}_k^\top \boldsymbol{r}_k = \boldsymbol{J}_{DLS_k}^\dagger \boldsymbol{r}_k \tag{26}$$

- Minimum damped least squares (MDLS) [15]:

$$\boldsymbol{d}_k = (\boldsymbol{J}_k^\top \boldsymbol{J}_k + \lambda_{MDLS}\boldsymbol{I})^{-1}\boldsymbol{J}_k^\top \boldsymbol{r}_k = \boldsymbol{J}_{MDLS_k}^\dagger \boldsymbol{r}_k$$
$$\lambda_{MDLS} = \begin{cases} \lambda_{\text{base}}/\sigma_{\min}, & \text{if} \quad \sigma_{\min} < \sigma_{\text{thresh}} \\ 0, & \text{otherwise} \end{cases} \tag{27}$$

- Gaussian damped least squares (GDLS) [17]:

$$\boldsymbol{d}_k = (\boldsymbol{J}_k^\top \boldsymbol{J}_k + \lambda_{GDLS}\boldsymbol{I})^{-1}\boldsymbol{J}_k^\top \boldsymbol{r}_k = \boldsymbol{J}_{GLS_k}^\dagger \boldsymbol{r}_k$$
$$\lambda_{GDLS_i} = \lambda_{\max} e^{-(\sigma_i/\sigma_{\text{thresh}})^2} \quad \forall i \in [0, \ldots, m] \tag{28}$$

- Levenberg-Marquardt method (LM) [54]:

$$\boldsymbol{d}_k = (\boldsymbol{J}_k^\top \boldsymbol{J}_k + \lambda_{LM}\boldsymbol{I})^{-1}\boldsymbol{J}_k^\top \boldsymbol{r}_k = \boldsymbol{J}_{LM_k}^\dagger \boldsymbol{r}_k$$
$$\lambda_{LM_k} = \begin{cases} \lambda_{LM_{k-1}}/\beta_{LM}, & \text{if} \quad ||\boldsymbol{r}_{\text{new}}||_2 < ||\boldsymbol{r}_{\text{prev}}||_2 \\ \beta_{\text{LM}}\lambda_{LM_{k-1}}, & \text{otherwise} \end{cases} \tag{29}$$

with $\lambda_{LM_0} = \lambda_{\text{init}}$.

The Jacobian pseudoinverse $J^\dagger$ is computed using Singular Value Decomposition (SVD). The Jacobian is decomposed into $\boldsymbol{J} = \boldsymbol{V}\Sigma\boldsymbol{U}^\top$, where $\boldsymbol{V}$ and $\boldsymbol{U}$ are orthogonal matrices, and $\boldsymbol{\Sigma}$ is a diagonal matrix containing singular values of $\boldsymbol{J}$. The pseudoinverse is then calculated as:

$$\boldsymbol{J}^\dagger = \boldsymbol{V}\tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{U}^\top \tag{30}$$

Here, $\tilde{\boldsymbol{\Sigma}}$ represents the diagonal matrix composed of modified singular values $\tilde{\sigma}$. For the pseudoinverse (PINV) method, the modified singular values $\tilde{\sigma}_i$ are given by:

$$\tilde{\sigma}_i = \frac{1}{\sigma_i} \tag{31}$$

For the Damped Least Squares (DLS) and Levenberg-Marquardt (LM) method, the damping factor λ is incorporated into the modified singular values as follows:

$$\tilde{\sigma}_i = \frac{\sigma_i}{\sigma_i^2 + \lambda^2}, \tag{32}$$

while the Minimum Damped Least Squares (MDLS) and Gaussian Damped Least Squares (GDLS) methods are incorporated as

$$\tilde{\sigma}_i = \frac{1}{\sigma_i + \lambda} \tag{33}$$

To ensure that the joint angles remain within the specified joint limits $[\underline{q_i}, \overline{q_i}]$, a clipping strategy is employed at the end of each iteration:

$$q_i = \begin{cases} \overline{q_i}, & \text{if } q_i > \overline{q_i} \\ \underline{q_i}, & \text{if } q_i < \underline{q_i} \end{cases} \tag{34}$$

This strategy is implemented to prevent the joint angles from exceeding their allowable range, ensuring the physical feasibility and operational safety of the robotic manipulator.

B. EVALUATED VARIABLE STEP SIZES

This study compares the performance of the proposed variable step sizes with other step size criteria commonly used in gradient-based optimization.

- Constant: The step size is consistently constant at 1.

$$\alpha_k^{\text{FIXED}} = 1 \tag{35}$$

- Steepest descent [55]: The step size is determined based on the steepest descent criterion.

$$\alpha_k^{\text{SD}} = \frac{\boldsymbol{r}_k^\top \boldsymbol{r}_k}{\boldsymbol{r}_k^\top \boldsymbol{J}_k \boldsymbol{J}_k^\top \boldsymbol{r}_k} \tag{36}$$

- Barzilai and Borwein BB1: This method calculates the step size according to the BB1 criterion.

$$\alpha_k^{\text{BB1}} = \frac{\boldsymbol{r}_{k-1}^\top \boldsymbol{r}_{k-1}}{\boldsymbol{r}_{k-1}^\top \boldsymbol{J}_{k-1} \boldsymbol{J}_{k-1}^\top \boldsymbol{r}_{k-1}} \tag{37}$$

- Barzilai and Borwein BB2: This approach uses the BB2 criterion to determine the step size.

$$\alpha_k^{\text{BB2}} = \frac{\boldsymbol{r}_{k-1}^\top \boldsymbol{J}_{k-1} \boldsymbol{J}_{k-1}^\top \boldsymbol{r}_{k-1}}{\boldsymbol{r}_{k-1}^\top \left(\boldsymbol{J}_{k-1} \boldsymbol{J}_{k-1}^\top\right)^2 \boldsymbol{r}_{k-1}} \tag{38}$$

- Gradient Scaling: The step size of each joint is scaled based on the norm of the gradient vector.

$$\alpha_{k_i}^{\text{GS}} = \frac{|\boldsymbol{d}_{k_i}|}{\|\boldsymbol{d}_k\|} \quad \forall i \in [1, \ldots .n] \tag{39}$$

- Relative Gradient Norm: The step size is scaled based on the Relative Gradient Norm,

$$\alpha_{k_i}^{\text{RGN}} = \frac{\varphi_{k_i}}{\|\varphi_k\|^2} \quad \forall i \in [1, \ldots .n] \tag{40}$$

where $\varphi_{k_i} = \left|\frac{d_{k_i}}{q_i}\right| \quad \forall i \in [1, \ldots .n]$.

- Cyclical: This method uses a cyclical pattern of step sizes that alternates through a predefined sequence in cycles,

$$\alpha_k^{ALT} = S_{(k \mod N)} \tag{41}$$

with $N = 7$ and $S = \{1/\phi^2, 1/\phi, 1.0, \phi, 1.0, 1/\phi, 1/\phi^2\}$, where $\phi = 1.618$ is the golden ratio.

- Uniform Random: The step size is randomly sampled from a uniform distribution.

$$\alpha^{\text{URAND}} \sim \mathcal{U}(0.5, 1.5) \tag{42}$$

- Normal Random: The step size is randomly sampled from a normal distribution.

$$\alpha^{\text{NRAND}} \sim \mathcal{N}(1.0, 0.5) \tag{43}$$

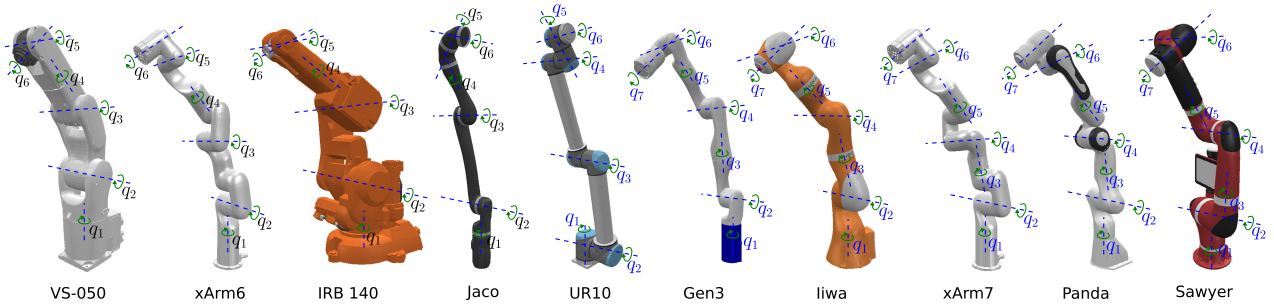

FIGURE 3. Illustration of the robotic manipulator models used for experimental validation.

C. EXPERIMENTAL SETUP

The proposed approach is evaluated on ten commercial manipulator kinematic chains, including both 6-DOF and 7-DOF manipulators. The manipulators were selected based on criteria that ensure a diverse representation of commonly used industrial robots, including factors such as wrist configuration, workspace range, and the presence of redundant DOFs. The 6-DOF manipulators evaluated are:

- UR-10 (Universal Robots)
- VS-050 (DENSO Robotics)
- xArm6 (UFactory)
- Jaco (Kinova)
- IRB140 (ABB)

The 7-DOF manipulators evaluated are:

- Gen3 (Kinova Robotics)
- Panda (Franka Emika)
- LBR iiwa (KUKA)
- xArm7 (UFactory)
- Sawyer (Rethink Robotics)

The evaluated kinematic chains are illustrated in Fig. 3. To assess the effectiveness of the proposed approach, 5000 target poses, each including an end-effector position and orientation, were randomly generated using forward kinematics applied to random joint configurations. This ensures that each target pose has at least one valid solution. Integrating 6D target poses introduces substantial difficulties for inverse kinematics algorithms, particularly in terms of reachability and the likelihood of facing singularities. Moreover, 5000 initial joint configurations were also randomly generated and paired with a target pose. The implementation was carried out in Python, employing the Pinocchio library (v. 2.6.10) [56] for kinematic computations, transformations, and kinematic chain parsing. The choice of damping factors and thresholds is not trivial, as they can significantly affect the performance of IK methods. For the experimental analysis, a predefined set of potential options was considered for each parameter. The complete list of parameters used for the experimental evaluation is summarized in Table 1.

D. COMPARATIVE ANALYSIS OF SOLVE RATES FOR VARIABLE STEP SIZES

This section presents a comparative analysis of the solve rates achieved by various step size methods. The solve rates for each variable step size method are summarized in Table 2. The methods are evaluated based on their ability to improve the solve rates when compared to a fixed step size approach. The descent direction for each method is derived using the Jacobian pseudoinverse algorithm (PINV). To quantify the relative effectiveness of each method, we calculate the relative solve rates, SR_{rel}, with respect to a constant step size ($\alpha^{\text{FIXED}} = 1.0$), as follows:

$$\text{SR}_{\text{rel}}(\alpha) = \frac{\text{SR}(\alpha) - \text{SR}(\alpha^{\text{FIXED}})}{\text{SR}(\alpha^{\text{FIXED}})} \times 100\% \qquad (44)$$

where $\text{SR}(\alpha)$ represents the solve rate for a given step size α. Figures 4A and 4B illustrate these relative solve rates for non-redundant and redundant kinematic chains, respectively.

The results show that a constant step size α^{FIXED} limits the solve rate for PINV, while variable step sizes offer varying degrees of solve rate improvement. For non-redundant kinematic chains, where a single solution is expected, the impact of variable step sizes is substantial, with relative solve rates exceeding 300% for the UR10 manipulator. In the case of redundant kinematic chains, the relative solve rate reaches upt to 35% for the Sawyer robot. Methods such as BB1, BB2, RGN, and NRAND demonstrate the most significant improvements in the solve rates for different kinematic chains. The strict residual reduction enforced by the SD, BB1, and BB2 methods appears to be effective for both types of kinematic chains. Among the proposed random-based variable step sizes, the NRAND method, which samples step sizes randomly from a normal distribution, achieved the best performance in half of the evaluated cases. In contrast, the URAND version, sampling from a uniform distribution, provided limited improvement. This observation suggests

TABLE 1. Parameters for experimental validation.

Parameter	Value
Residual function ($\mathbf{r}(\mathbf{q})$)	$\log(\mathbf{X}_d\mathbf{X}_a^{-1})$
Convergence threshold (ϵ_c)	1×10^{-5}
Max. number of iterations	5000
λ_{DLS}	$\{0.1, 0.01, 0.001\}$
λ_{base}	$\{10^{-3}, 10^{-4}, 10^{-5}\}$
$\lambda_{\max}$	$\{0.1, 0.01, 0.001\}$
λ_{init}	$\{0.1, 0.01, 0.001\}$
σ_{thresh}	0.01
β_{LM}	2

TABLE 2. Comparison of solve rates across different step size methods. The highest solve rates for each kinematic chain are emphasized in bold.

Kin. Chain	α^{FIXED}	α^{SD}	α^{BB1}	α^{BB2}	α^{RGN}	α^{GS}	α^{ALT}	α^{URAND}	α^{NRAND}
VS-050	0.262	0.281	0.289	0.292	0.268	0.253	0.275	0.261	**0.293**
xArm6	0.177	0.314	0.324	**0.339**	0.231	0.234	0.225	0.184	0.304
IRB140	0.113	0.141	0.144	0.144	0.093	0.101	0.118	0.116	**0.165**
Jaco	0.065	0.240	0.244	**0.277**	0.157	0.173	0.102	0.067	0.063
UR10	0.091	0.261	0.295	0.361	0.142	0.159	0.157	0.139	**0.416**
Gen3	0.585	0.602	**0.603**	0.600	0.592	0.585	0.586	0.586	0.583
Iiwa	0.443	0.444	0.442	0.443	**0.453**	0.439	0.445	0.436	0.450
xArm7	0.470	0.544	**0.547**	0.546	0.496	0.510	0.503	0.479	0.511
Panda	0.363	0.386	0.385	0.383	0.341	0.365	0.369	0.371	**0.448**
Sawyer	0.396	0.538	0.540	0.543	0.455	0.486	0.466	0.411	**0.548**

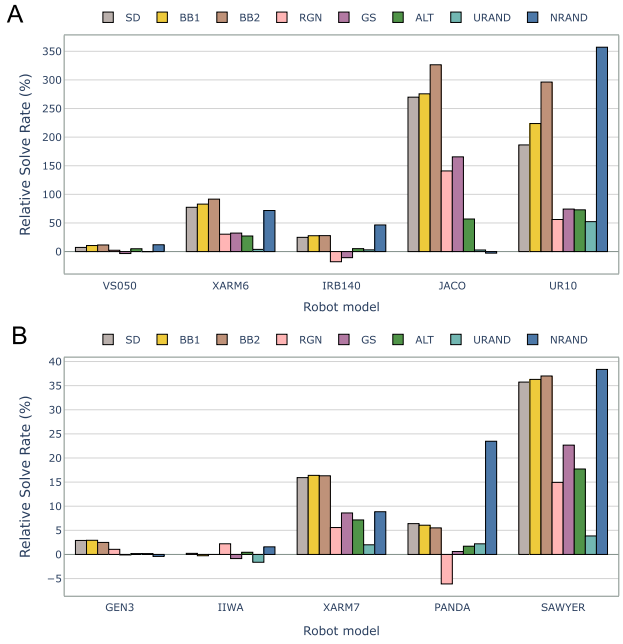

FIGURE 4. Comparison of relative solve rates SR_{rel}, expressed as the percentage change in solve rate relative to a constant step size ($\alpha = 1.0$). A. Non-redundant kinematic chains. B. Redundant kinematic chains.

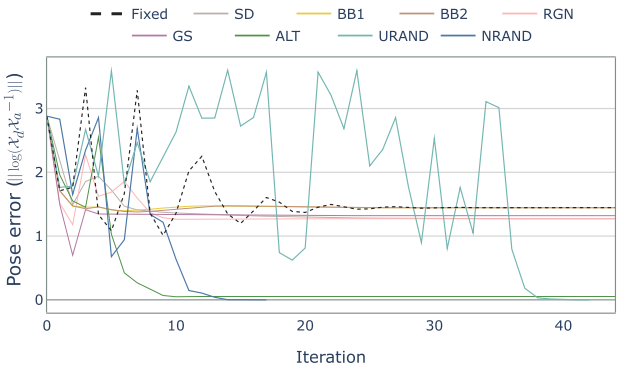

FIGURE 5. Comparison of pose error reduction across iterations for different step size methods in Jacobian pseudoinverse IK.

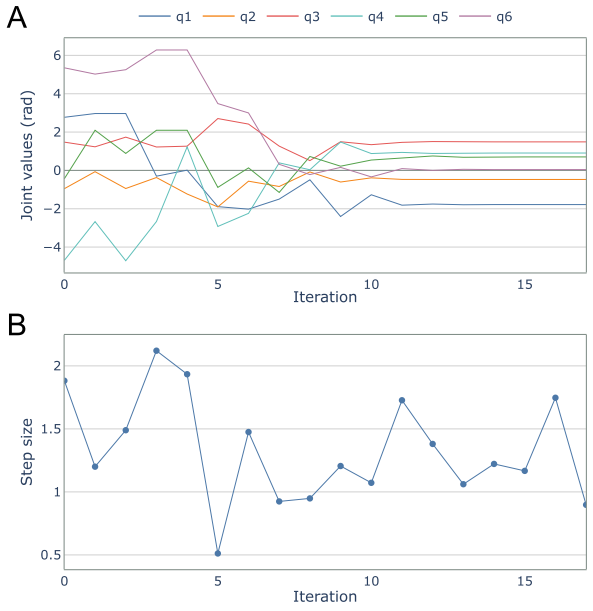

FIGURE 6. Visualization of IK optimization dynamics: A. Evolution of joint values across iterations. B. Corresponding step sizes, randomly sampled from a normal distribution, used in each iteration.

that maintaining step sizes around the default value of $\alpha = 1.0$ ensures a more focused exploration of the solution space, leading to more efficient convergence, while still reducing the risk of getting stuck in local minima.

E. CASE STUDY

A case study is presented using one of the randomly generated initial and target poses to demonstrate the effect of variable step sizes in inverse kinematics problems. We selected the VS-050 kinematic chain with initial configuration $\boldsymbol{q}_0 = [2.771, -0.952, 1.475, -4.690, -0.430, 5.352]$. The target pose within the reachable workspace is defined as

$$\boldsymbol{\mathcal{X}}_d = \begin{bmatrix} 0.679 & 0.661 & 0.317 & 0.0016 \\ 0.181 & 0.268 & -0.946 & -0.1617 \\ -0.711 & 0.700 & 0.0626 & 0.7148 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Figure 5 shows the error between the end-effector pose and the target, computed after (23), between optimization iterations. Methods such as SD, BB1, and BB2 continuously reduce the error, only increasing when joint limits are reached and the solution is truncated. However, they were unable to find a solution within the convergence threshold. Gradient-based step sizes, RGN and GS, rapidly reduce the error but get

TABLE 3. Comparison of solve rates with random restarts for different step size methods. The highest solve rates for each kinematic chain are emphasized in bold.

Kin. Chain	α^{FIXED}	α^{SD}	α^{BB1}	α^{BB2}	α^{RGN}	α^{GS}	α^{ALT}	α^{URAND}	α^{NRAND}
VS-050	0.612	0.659	0.608	0.612	0.662	0.663	0.657	0.664	**0.682**
xArm6	0.628	0.782	0.634	0.627	0.768	0.759	0.782	0.792	**0.839**
IRB140	0.356	0.398	0.355	0.353	0.424	0.436	0.435	0.470	**0.550**
Jaco	0.175	0.339	0.171	0.171	0.371	0.395	0.353	0.393	**0.437**
UR10	0.324	0.421	0.319	0.323	0.862	0.880	0.540	0.910	**0.939**
Gen3	0.677	0.643	0.677	0.677	0.661	0.668	0.675	0.677	**0.678**
Iiwa	0.707	0.556	0.707	0.709	0.700	0.696	0.715	0.720	**0.728**
xArm7	0.817	0.739	0.816	0.815	0.816	0.823	0.849	0.850	**0.860**
Panda	0.719	0.505	0.711	0.715	0.659	0.716	0.757	0.774	**0.813**
Sawyer	0.818	0.746	0.823	0.817	0.870	0.878	0.903	0.914	**0.932**

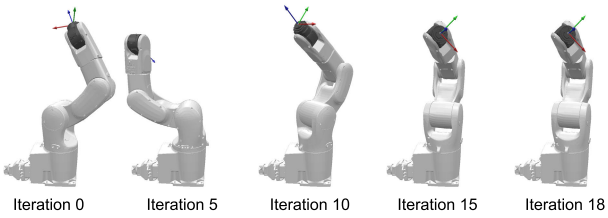

FIGURE 7. Sequential snapshots illustrating the manipulator's configurations at different iterations during the inverse kinematics optimization process.

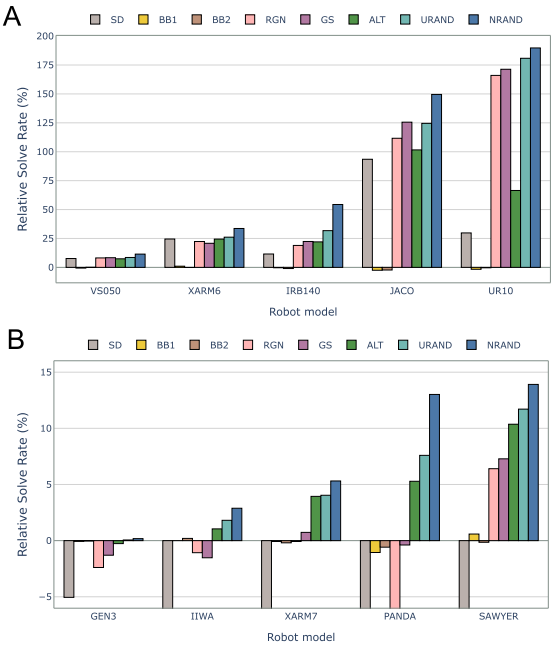

FIGURE 8. Comparison of relative solve rates SR_{rel} with random restarts, expressed as the percentage change in solve rate relative to a constant step size ($\alpha^{\text{FIXED}} = 1.0$). A. Non-redundant kinematic chains. B. Redundant kinematic chains.

stuck in a local minimum upon reaching a joint limit. Cyclic alternation (ALT) achieves error reduction close to the desired threshold but does not converge fully. In contrast, random step size approaches, URAND and NRAND, not strictly adhering to an error-reducing criterion, successfully found the optimal solution in less than 50 iterations.

Figure 6A shows the joint angles for the most effective method, NRAND, at each iteration, alongside the step size randomly sampled as shown in Fig. 6B. A total of 18 iterations were required to achieve convergence. Snapshots of robot configurations at various iterations are shown in Fig. 7. Although some joints also reached their limits at certain points, they did not get stuck due to the random nature of the step size selection, alternating between large values in the initial iterations, shorter values around iteration 5, and then sizes close to 1.0 in iterations 7-9. Once the joint configuration approached the optimal solution in iteration 10, the subsequent steps remained smooth, and the size of the steps did not hinder convergence.

F. RANDOM RESTARTS

Gradient-based methods are prone to get stuck in local minima. To address this challenge and further enhance the solve rate, we evaluated the proposed step sizes in conjunction with an improved random restart approach. This technique allows the initial joint configuration to be reset with random values after a predetermined number of iterations without a significant reduction in pose error. A threshold of 1×10^{-6} for pose error and a limit of 10 iterations without improvement were set to trigger a restart.

The random restart mechanism offers a systematic way to escape local minima and explore different regions of the solution space. By periodically restarting the optimization process, the likelihood of finding the global minimum for the inverse kinematics problem is increased.

The solve rates with random restarts for each evaluated kinematic chain and step size selection method are presented in Table 3, while the comparison of relative solve rates is depicted in Fig. 8. While Section III-D revealed relative improvements in solve rates, the overall rates remained below 50% for most cases. The introduction of random restarts significantly increased the solve rates for all robots, with most achieving solve rates greater than 50%. The Jaco robot was an exception due to its unique wrist arrangement. Variable step sizes continue to play a crucial role in enhancing solve rates. SD, BB1, and BB2 methods did not show advantages when combined with random restarts. Gradient-based step sizes like RGN and GS displayed significant improvement for non-redundant kinematic chains, but less so for 7-DOF

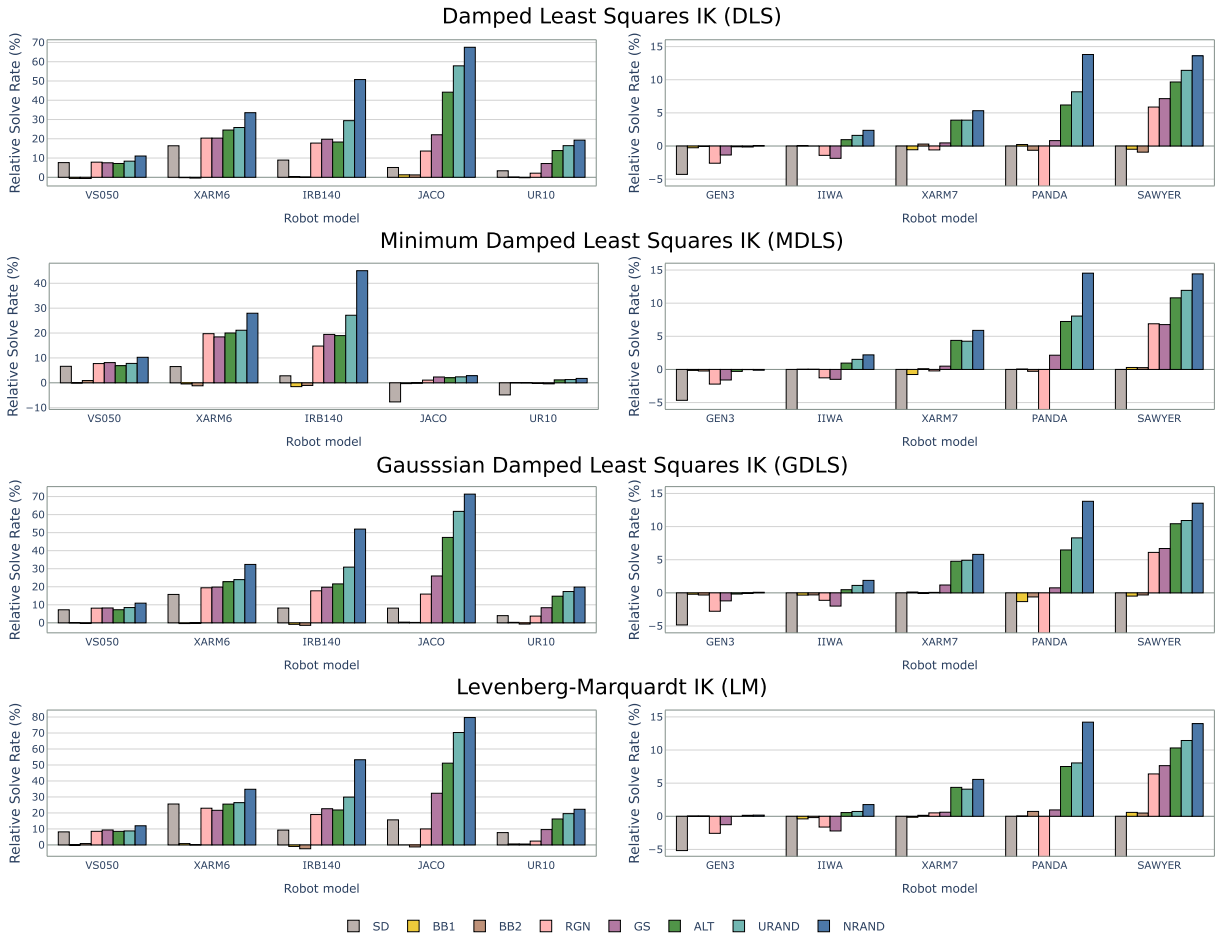

FIGURE 9. Relative solve rates SR_{rel} across different kinematic chains using damped inverse kinematics algorithms with various step size selections.

TABLE 4. Average solve rates across all non-redundant kinematic chains for various IK algorithms with variable step sizes. Top performing rates for each algorithm are highlighted in bold.

IK Algorithm	α^{FIXED}	α^{SD}	α^{BB1}	α^{BB2}	α^{RGN}	α^{GS}	α^{ALT}	α^{URAND}	α^{NRAND}
PINV	0.419	0.520	0.417	0.417	0.617	0.627	0.553	0.646	**0.690**
DLS	0.569	0.615	0.569	0.568	0.632	0.648	0.682	0.709	**0.751**
MDLS	0.663	0.662	0.661	0.662	0.711	0.714	0.717	0.727	**0.754**
GDLS	0.567	0.614	0.566	0.564	0.633	0.651	0.684	0.710	**0.751**
LM	0.559	0.632	0.560	0.558	0.623	0.655	0.684	0.713	**0.756**

TABLE 5. Average solve rates across all redundant kinematic chains for various IK algorithms with variable step sizes. Top performing rates for each algorithm are highlighted in bold.

IK Algorithm	α^{FIXED}	α^{SD}	α^{BB1}	α^{BB2}	α^{RGN}	α^{GS}	α^{ALT}	α^{URAND}	α^{NRAND}
PINV	0.748	0.638	0.747	0.746	0.741	0.756	0.780	0.787	**0.802**
DLS	0.747	0.631	0.746	0.745	0.739	0.756	0.779	0.786	**0.801**
MDLS	0.743	0.612	0.742	0.743	0.736	0.754	0.779	0.783	**0.800**
GDLS	0.746	0.631	0.743	0.744	0.738	0.756	0.781	0.785	**0.800**
LM	0.746	0.638	0.746	0.748	0.743	0.756	0.782	0.784	**0.801**

chains. In contrast, cyclic alternation (ALT) and random approaches (URAND and NRAND) significantly improved solve rates across all kinematic chains. In particular, the NRAND method achieved the highest improvements for all robot models, with solve rates exceeding 90% for the UR10 and Sawyer kinematic chains and showing substantial increases in all other chains.

G. EVALUATION OF VARIABLE STEP SIZES WITH VARIOUS INVERSE KINEMATICS ALGORITHMS

The performance of variable step sizes was evaluated with various inverse kinematics algorithms, including DLS, MDLS, GDLS, and LM. As these algorithms depend on the selection of appropriate damping factors and additional parameters, multiple combinations were tested, as indicated

TABLE 6. Comparison of average iterations required for convergence with various step sizes across different IK algorithms.

IK Algorithm	α^{FIXED}	α^{SD}	α^{BB1}	α^{BB2}	α^{RGN}	α^{GS}	α^{ALT}	α^{URAND}	α^{NRAND}
PINV	200.2	916.5	195.6	196.3	618.1	652.1	214.8	305.1	364.5
DLS	154.9	915.2	155.5	156.0	689.6	706.1	186.6	211.5	244.9
MDLS	166.5	893.9	168.4	165.3	617.0	640.1	184.7	211.3	236.3
GDLS	154.3	908.8	155.7	153.8	683.5	704.5	186.2	214.9	242.6
LM	157.1	930.2	154.9	155.2	693.0	709.9	188.9	216.9	245.1

TABLE 7. Average computation time in seconds for each IK algorithm across different step sizes.

IK Algorithm	α^{FIXED}	α^{SD}	α^{BB1}	α^{BB2}	α^{RGN}	α^{GS}	α^{ALT}	α^{URAND}	α^{NRAND}
PINV	0.030	0.133	0.027	0.027	0.098	0.099	0.032	0.040	0.047
DLS	0.020	0.125	0.021	0.021	0.093	0.093	0.027	0.023	0.027
MDLS	0.021	0.140	0.026	0.026	0.084	0.083	0.029	0.030	0.033
GDLS	0.016	0.141	0.022	0.022	0.075	0.075	0.027	0.027	0.030
LM	0.020	0.154	0.024	0.023	0.091	0.090	0.030	0.026	0.029

in Table 1. The best results obtained from all combinations are reported in this section. The relative solve rate for each kinematic chain, step size, and IK method is shown in Fig. 9. Regardless of the chosen IK approach, a performance trend similar to that observed with the PINV method is found with the damped IK approaches. In particular, NRAND exhibited the best performance among all variable step sizes, enhancing solve rates for all kinematic chains. This demonstrates the effectiveness of randomly selecting step sizes within a normal distribution centered on $\alpha = 1.0$.

Tables 4 and 5 present the average solve rates for non-redundant and redundant kinematic chains, respectively. Among non-redundant robots, the Levenberg-Marquardt method achieves the best performance, significantly benefiting from variable steps, as reflected in the 80% relative solve rate for the JACO robot. On the contrary, PINV shows the lowest average solve rate, while MDLS, already largely benefiting from random restarts, receives the least advantage from variable steps. For 7-DOF chains, most IK methods reflect a similar trend, with slight variations in average solve rates. The NRAND method achieves the best performance, followed by the URAND approach. Generally, all IK approaches experience an approximate 9% improvement in the solve rates when combined with NRAND. In contrast, SD, BB1, and BB2 methods either maintain or significantly reduce solve rate performance, likely due to a high probability of becoming stuck in local minima at each restart. Random step size methods are more effective in avoiding this issue.

H. COMPUTATIONAL PERFORMANCE

The computational performance was evaluated in terms of the number of iterations and computational time required for successful runs. Table 6 summarizes the average number of optimization iterations needed to find a solution across all kinematic chains. For the constant case, DLS required the fewest iterations, while PINV required the most. The impact of variable step sizes varied, with SD having the highest average number of iterations and BB1 the lowest. Cyclic alternation and random approaches increased the number of iterations by approximately 50% to 80% compared to the baseline case. However, this additional computational demand could be offset by the solve rate improvement demonstrated in previous sections.

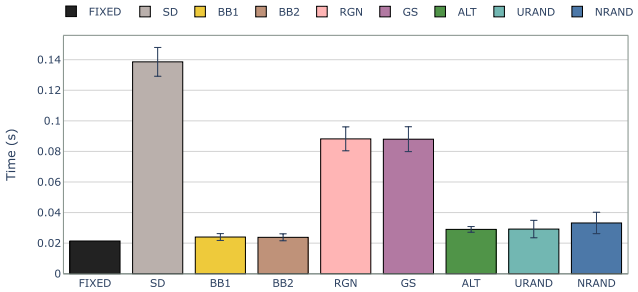

FIGURE 10. Average computation time in seconds across all robot kinematic chains for different step size strategies.

Table 7 presents the corresponding average computation time for each step size and IK algorithm, and Figure 10 shows the average computation time for each step size when considering all the IK algorithms evaluated. The best average computation time under constant step sizes was achieved with GDLS and LM approaches, around 20 ms, while PINV exhibited the highest time, about 30 ms. The use of randomly selected step sizes increased the average computation time by approximately 50%, with times ranging from 27 to 47ms. Nevertheless, given the potential for further optimization using faster compilers (e.g., C++), it is expected that a suitable performance for real-time control (>500Hz) can be readily achieved.

IV. CONCLUSION

This study has explored the impact of step size selection for Jacobian-based inverse kinematics (IK) methods for robotic manipulators. Although constant step size approaches are common, they often face limitations in convergence speed and overall efficiency. To overcome these drawbacks, we proposed and evaluated variable step size strategies that include gradient-based dynamic selection, cyclic alternation,

and random sampling. Our experimental analysis, conducted on various manipulator kinematic chains with 6 and 7 degrees of freedom, demonstrated the significant advantages of variable step sizes. Furthermore, we extended our evaluation to include various iterative Jacobian-based IK methods beyond the conventional Jacobian pseudoinverse, such as Damped Least Squares (DLS), Minimum Damped Least Squares (MDLS), Gaussian Damped Least Squares (GDLS), and Levenberg-Marquardt (LM) approaches. Our results showed a substantial improvement in solve rates achieved by employing random step sizes, particularly those derived from a normal distribution. This enhancement was further enhanced by integrating random restarts. Future work will aim to adapt these findings to multiobjective IK scenarios, where precise parameter selection is crucial for performance. We also plan to refine and enhance our approach to improve its efficiency and robustness for practical robotic applications. Additionally, we will further analyze computational complexity, stability of convergence, and sensitivity to step size parameters. Variable step sizes are applicable in numerous sectors to boost both efficiency and accuracy. For mobile robot path planning, these step sizes can facilitate real-time modifications in response to environmental changes. Within machine learning, especially during the training of deep neural networks, employing variable step sizes can accelerate convergence and help bypass local minima in gradient descent processes. Moreover, these techniques can be combined with optimization methods such as line search or trust region strategies to dynamically fine-tune step sizes. This study demonstrates that employing variable step sizes can lead to more effective and reliable outcomes in Jacobian-based inverse kinematics for robotic manipulators.

REFERENCES

[1] B. Siciliano, O. Khatib, and T. Kröger, *Springer Handbook of Robotics*. Cham, Switzerland: Springer, 2008.

[2] J. Colan, A. Davila, Y. Zhu, T. Aoyama, and Y. Hasegawa, "OpenRST: An open platform for customizable 3D printed cable-driven robotic surgical tools," *IEEE Access*, vol. 11, pp. 6092–6105, 2023.

[3] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. thesis, Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, 2010.

[4] S. Starke, N. Hendrich, and J. Zhang, "Memetic evolution for generic full-body inverse kinematics in robotics and animation," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 406–420, Jun. 2019.

[5] K. Fozilov, J. Colan, K. Sekiyama, and Y. Hasegawa, "Toward autonomous robotic minimally invasive surgery: A hybrid framework combining task-motion planning and dynamic behavior trees," *IEEE Access*, vol. 11, pp. 91206–91224, 2023.

[6] K. Fozilov, J. Colan, A. Davila, K. Misawa, J. Qiu, Y. Hayashi, K. Mori, and Y. Hasegawa, "Endoscope automation framework with hierarchical control and interactive perception for multi-tool tracking in minimally invasive surgery," *Sensors*, vol. 23, no. 24, p. 9865, Dec. 2023.

[7] H. Dai, G. Izatt, and R. Tedrake, "Global inverse kinematics via mixed-integer convex optimization," *Int. J. Robot. Res.*, vol. 38, nos. 12–13, pp. 1420–1441, Oct. 2019.

[8] K. Gupta and K. Kazerounian, "Improved numerical solutions of inverse kinematics of robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, Sep. 1985, pp. 743–748.

[9] P. Beeson and B. Ames, "TRAC-IK: An open-source library for improved solving of generic inverse kinematics," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, Nov. 2015, pp. 928–935.

[10] J. Colan, J. Nakanishi, T. Aoyama, and Y. Hasegawa, "Optimization-based constrained trajectory generation for robot-assisted stitching in endonasal surgery," *Robotics*, vol. 10, no. 1, p. 27, Feb. 2021.

[11] J. Colan, A. Davila, K. Fozilov, and Y. Hasegawa, "A concurrent framework for constrained inverse kinematics of minimally invasive surgical robots," *Sensors*, vol. 23, no. 6, p. 3328, Mar. 2023.

[12] I. Duleba and M. Opałka, "A comparison of jacobian-based methods of inverse kinematics for serial robot manipulators," *Int. J. Appl. Math. Comput. Sci.*, vol. 23, no. 2, pp. 373–382, Jun. 2013.

[13] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy," *Int. J. Robot. Res.*, vol. 10, no. 4, pp. 410–425, Aug. 1991.

[14] D. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man Mach. Syst.*, vols. MMS-10, no. 2, pp. 47–53, Jun. 1969.

[15] C. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Trans. Syst. Man, Cybern.*, vols. SMC-16, no. 1, pp. 93–101, Jan. 1986.

[16] S. R. Buss and J.-S. Kim, "Selectively damped least squares for inverse kinematics," *J. Graph. Tools*, vol. 10, no. 3, pp. 37–49, Jan. 2005.

[17] L. M. Phuoc, P. Martinet, S. Lee, and H. Kim, "Damped least square based genetic algorithm with Gaussian distribution of damping factor for singularity-robust inverse kinematics," *J. Mech. Sci. Technol.*, vol. 22, no. 7, pp. 1330–1338, Jul. 2008.

[18] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn.*, vol. 8, nos. 3–4, pp. 231–357, 2015.

[19] M. Raydan and B. F. Svaiter, "Relaxed steepest descent and cauchy-barzilai-borwein method," *Comput. Optim. Appl.*, vol. 21, pp. 155–167, Aug. 2002.

[20] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[21] J. M. Altschuler and P. A. Parrilo, "Acceleration by stepsize hedging I: Multi-step descent and the silver stepsize schedule," 2023, *arXiv:2309.07879*.

[22] A. Cauchy, "Méthode générale pour la résolution des systemes déquations simultanées," *Comp. Rend. Sci. Paris*, vol. 25, no. 1847, pp. 536–538, 1847.

[23] J. Barzilai and J. M. Borwein, "Two-point step size gradient methods," *IMA J. Numer. Anal.*, vol. 8, no. 1, pp. 141–148, 1988.

[24] Y. Yuan, "A new stepsize for the steepest descent method," *J. Comput. Math.*, vol. 24, no. 2, pp. 149–156, 2006.

[25] Y. Dai and Y. Yuan, "Analysis of monotone gradient methods," *J. Ind. Manag. Optim.*, vol. 1, no. 2, pp. 181–192, Jul. 2005.

[26] B. Zhou, L. Gao, and Y.-H. Dai, "Gradient methods with adaptive step-sizes," *Comput. Optim. Appl.*, vol. 35, no. 1, pp. 69–86, Sep. 2006.

[27] R. D. Asmundis, D. D. Serafino, W. W. Hager, G. Toraldo, and H. Zhang, "An efficient gradient method using the yuan steplength," *Comput. Optim. Appl.*, vol. 59, no. 3, pp. 541–563, Dec. 2014.

[28] Z. Liu, H. Liu, and X. Dong, "An efficient gradient method with approximate optimal stepsize for the strictly convex quadratic minimization problem," *Optimization*, vol. 67, no. 3, pp. 427–440, Mar. 2018.

[29] Y.-K. Huang, Y.-H. Dai, and X.-W. Liu, "Equipping the barzilai-borwein method with the two dimensional quadratic termination property," *SIAM J. Optim.*, vol. 31, no. 4, pp. 3068–3096, Jan. 2021.

[30] Y. Nesterov, "Gradient methods for minimizing composite functions," *Math. Program.*, vol. 140, no. 1, pp. 125–161, Aug. 2013.

[31] T. T. Truong and H.-T. Nguyen, "Backtracking gradient descent method and some applications in large scale optimisation. Part 2: Algorithms and experiments," *Appl. Math. Optim.*, vol. 84, no. 3, pp. 2557–2586, Dec. 2021.

[32] L. Armijo, "Minimization of functions having Lipschitz continuous first partial derivatives," *Pacific J. Math.*, vol. 16, no. 1, pp. 1–3, Jan. 1966.

[33] C. Paquette and K. Scheinberg, "A stochastic line search method with convergence rate analysis," 2018, *arXiv:1807.07994*.

[34] A. S. Berahas, L. Cao, K. Choromanski, and K. Scheinberg, "A theoretical and empirical comparison of gradient approximations in derivative-free optimization," 2019, *arXiv:1905.01332*.

[35] S. Vaswani, A. Mishkin, I. Laradji, M. Schmidt, G. Gidel, and S. Lacoste-Julien, "Painless stochastic gradient: Interpolation, line-search, and convergence rates," 2019, *arXiv:1905.09997*.

[36] J. Bernstein, C. Mingard, K. Huang, N. Azizan, and Y. Yue, "Automatic gradient descent: Deep learning without hyperparameters," 2023, *arXiv:2304.05187*.
[37] Y. Lee, A. S. Chen, F. Tajwar, A. Kumar, H. Yao, P. Liang, and C. Finn, "Surgical fine-tuning improves adaptation to distribution shifts," 2023, *arXiv:2210.11466*.
[38] B. Goujaud, D. Scieur, A. Dieuleveut, A. Taylor, and F. Pedregosa, "Super-acceleration with cyclical step-sizes," 2021, *arXiv:2106.09687*.
[39] B. Grimmer, "Provably faster gradient descent via long steps," 2023, *arXiv:2307.06324*.
[40] M. Schumer and K. Steiglitz, "Adaptive step size random search," *IEEE Trans. Autom. Control*, vols. AC-13, no. 3, pp. 270–276, Jun. 1968.
[41] L. Blier, P. Wolinski, and Y. Ollivier, "Learning with random learning rates," in *Machine Learning and Knowledge Discovery in Databases*. Cham, Switzerland: Springer, 2020, pp. 449–464.
[42] X. Wang, J. Cao, X. Liu, L. Chen, and H. Hu, "An enhanced step-size Gaussian damped least squares method based on machine learning for inverse kinematics of redundant robots," *IEEE Access*, vol. 8, pp. 68057–68067, 2020.
[43] X. Wang, X. Liu, L. Chen, and H. Hu, "Deep-learning damped least squares method for inverse kinematics of redundant robots," *Measurement*, vol. 171, Feb. 2021, Art. no. 108821.
[44] N. Gould, D. Orban, and P. Toint, "Numerical methods for large-scale nonlinear optimization," *Acta Numerica*, vol. 14, pp. 299–361, May 2005.
[45] T. Sugihara, "Solvability-unconcerned inverse kinematics by the Levenberg–Marquardt method," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 984–991, Oct. 2011.
[46] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Philadelphia, PA, USA: SIAM, 1996.
[47] A. S. Deo and I. D. Walker, "Overview of damped least-squares methods for inverse kinematics of robot manipulators," *J. Intell. Robotic Syst.*, vol. 14, no. 1, pp. 43–68, Sep. 1995.
[48] R. De Asmundis, D. di Serafino, F. Riccio, and G. Toraldo, "On spectral properties of steepest descent methods," *IMA J. Numer. Anal.*, vol. 33, no. 4, pp. 1416–1435, Oct. 2013.
[49] A. Davila, J. Colan, and Y. Hasegawa, "Comparison of fine-tuning strategies for transfer learning in medical image classification," *Image Vis. Comput.*, vol. 146, Jun. 2024, Art. no. 105012.
[50] J. Colan, A. Davila, and Y. Hasegawa, "Enhancing gradient-based inverse kinematics with dynamic step sizes," in *Proc. Int. Symp. Micro-Nano Mehatronics Hum. Sci. (MHS)*, Nov. 2023, pp. 1–6.
[51] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Math. Oper. Res.*, vol. 6, no. 1, pp. 19–30, Feb. 1981.
[52] G. Schrack and M. Choit, "Optimized relative step size random searches," *Math. Program.*, vol. 10, no. 1, pp. 230–244, Dec. 1976.
[53] J. Sola, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," 2018, *arXiv:1812.01537*.
[54] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *J. Soc. Ind. Appl. Math.*, vol. 11, no. 2, pp. 431–441, Jun. 1963.
[55] T. Sugihara, "Solvability-unconcerned inverse kinematics based on Levenberg–Marquardt method with robust damping," in *Proc. 9th IEEE-RAS Int. Conf. Humanoid Robots*, Dec. 2009, pp. 555–560.
[56] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The pinocchio C++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Jan. 2019, pp. 614–619.

JACINTO COLAN (Member, IEEE) received the B.S. degree from the National University of Engineering, Lima, Peru, in 2010, and the M.E. and Ph.D. degrees in micro-nano mechanical science and engineering from Nagoya University, Nagoya, Japan, in 2018 and 2021, respectively. He is currently a Postdoctoral Researcher with Nagoya University. His main research interests include medical robotics, human–robot interfaces, and intelligent assistive systems. He received several conference awards, including the IEEE RO-MAN 2023, the IEEE CBS 2022, and the IEEE MHS 2022 Best Paper Awards.

ANA DAVILA (Member, IEEE) received the B.E. degree from the National University of San Antonio Abad in Cusco, Cusco, Peru, in 2009, the M.S. degree in life science from Kyoto University, Japan, in 2018, and the Ph.D. degree in medical sciences from Osaka University, Japan, in 2022. She is currently a Postdoctoral Researcher with Nagoya University, Japan. Her main research interests include intelligent systems, human–robot interaction, medical robotics, and bioinformatics.

YASUHISA HASEGAWA (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in robotics from Nagoya University, Nagoya, Japan, in 1994, 1996, and 2001, respectively. From 1996 to 1998, he was with Mitsubishi Heavy Industries Ltd., Japan. He joined Nagoya University, in 1998. He moved to Gifu University, in 2003. From 2004 to 2014, he was with the University of Tsukuba. Since 2014, he has been with Nagoya University, where he is currently a Professor with the Department of Micro-Nano Systems Engineering. His main research interests include the research fields of motion-assistive systems, teleoperation for manipulation, and surgical support robot.